VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SBĚR A VIZUALIZACE DAT ZE SIMULÁTORU ŘÍZENÍ VOZIDLA
COLLECTION AND VISUALIZATION OF DATA FROM THE VEHICLE DRIVING SIMULATOR

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Dominik Viater**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. David Michalík**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Dominik Viater ***ID:*** 204765

Ročník: 3 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Sběr a vizualizace dat ze simulátoru řízení vozidla

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se se softwarovým vybavením simulátoru řízení vozidla na UAMT
2. Proveďte rešerši v oblasti webových serverů a databází pro Windows Server
3. Realizujte ukládání dat ze simulátoru do databáze
4. Navrhněte webovou aplikaci pro vizualizaci měřených dat ze simulátoru s použitím dynamických prvků (JavaScript, PHP)
5. Ověřte správnost naměřených a zobrazených dat

DOPORUČENÁ LITERATURA:

RICHARDSON, Leonard. RESTful Web APIs. 2013. ISBN 9781449359744.

Termín zadání: 8.2.2021 ***Termín odevzdání:*** 24.5.2021

Vedoucí práce: Ing. David Michalík

doc. Ing. Václav Jirsík, CSc.
předseda rady studijního programu

ABSTRAKT

Táto bakalárska práca sa venuje zberu a vizualizácii dát zo simulátoru jazdy vozidlom. V teoretickom úvode sa zaoberá popisom herného enginu Unreal engine 4 a jeho editoru, rešerši webových serverov a rešerši open source databáz. V praktickej časti sa venuje zberu dát zo simulácie, porovnaniu dvoch typov zberu dát, a následne vizualizácii týchto dát.

KĽÚČOVÉ SLOVÁ

Unreal engine 4, herný engine, simulátor, webový server, databáza, MySQL, Apache, zber dát, vizualizácia dát

ABSTRACT

In this bachelor thesis, I deal with the collection and visualization of data from a vehicle driving simulator. In the theoretical introduction, I deal with the description of the game engine Unreal Engine 4 and its editor, web server search, open-source database search. The practical part deals with data collection from simulation, comparison of two types of data collection, and then visualization of this data.

KEYWORDS

Unreal engine 4, game engine, simulator, web server, database, MySQL, Apache, data collection, data visualization

VIATER, Dominik. *Sběr a vizualizace dat ze simulátoru řízení vozidla*. Brno, 2021, 52 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedúci práce: Ing. David Michalík

VYHLÁSENIE

Vyhlasujem, že svoju bakalársku prácu na tému „Sběr a vizualizace dat ze simulátoru řízení vozidla“ som vypracoval samostatne pod vedením vedúceho bakalárskej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Davidovi Michalíkovi za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora

Obsah

Zoznam obrázkov

Úvod

Cieľom tejto bakalárskej práce je vhodne uložiť dáta zo simulácie jazdy vozidlom a tieto dáta zobraziť. V teoretickom úvode práca popisuje Unreal Enginu 4, rešerš webových serverov a databáz. potrebných pre vývoj aplikácie V kapitole 1 je predstavený Unreal Enigine 4, v ktorom budú simulované dáta a taktiež detailne načrtne Unreal editor, čo je vývojové prostredie pre UE4. Kapitola 2 sa zaoberá rešeršou webových serverov a porovnaním dvoch najpoužívanejších open-source webových serverov, na základe ktorých bude vybraný správny server pre túto prácu. Pri rešerši databáz v kapitole 3 budú ukázané rôzne typy databáz a ich hlavné výhody oproti ostatným. Pri simulácii jazdy vozidlom je potrebné zabezpečiť požadovaný hardvér a softvér, aby simulácia odrážala reálne podmienky, o čom sa píše v štvrtej kapitole. Opíšaná bude dráha simulácie, ktorá slúži ako testovacia.

Praktická časť projektu je zameraná na spôsob ukladania a zobrazenia nameraných dát. Pri zbere dát je potrebné zvoliť správny formát, v ktorom sa budú dáta ukladať. Kapitola 5 načrtne save menu, dátový formát a funkcie, pomocou ktorých sa dáta ukladajú do súboru. K týmto dátam je niekedy problematický prístup a preto je potrebné vytvoriť aj variant online. Na prekonanie tejto prekážky sa využíva zber dát do databázy popísaný v kapitole 5.2. Zároveň je súčasťou tejto kapitoly aj porovnanie spomenutých spôsobov ukladania.

Po zozbieraní simulovaných dát je potrebné ich aj správne zobraziť, aby bolo možné vykonať analýzu. V súčastnosti je už takmer všetko online, a preto by bolo vhodné aby dáta tiež boli zobrazené online. Tento spôsob zobrazenia je načrtnutý v kapitole 6

1 Herný Engine

Za posledné desaťročie bolo vytvorených veľa herných enginov. Existujú enginy na distribúciu projektov (simulácii, hier a pod.) na prakticky ľubovoľnú platformu, od konzol a počítačov po mobilné a vreckové zariadenia. K dispozícii sú herné enginy pre každú úroveň zručností od tých, ktoré sú určené pre začiatočníkov (GameMaker Studio 2), až po pokročilejšie špičkové nástroje (Lumberyard, Unreal Engine).[1] Poskytujú funkcie od animácie po umelú inteligenciu. Herné enginy sú zodpovedné za vykreslenie grafiky, detekciu kolízií, správu pamäte a mnoho ďalších možností. Pomáhajú vývojárom pri programovaní a to tak, že navrhnuté enginy môžu použiť v iných projektoch. Engine pomáha oživiť postavy zo simulácie tým, že pomáha pri tvorbe scén, postáv a 3D animovanej grafiky, zvuku, umelej inteligencie, skriptovacej animácie, atď. Je ako integrované vývojové prostredie s pripravenou sadou nástrojov vizuálneho vývoja a opakovane použiteľnými softvérovými komponentmi. Zjednodušuje zložitú úlohu vývoja rôznych projektov poskytnutím abstrakčnej vrstvy, vďaka ktorej veľa veľkých úloh vyzerá jednoducho, zatiaľ čo engine robí všetku prácu na pozadí.[2]

Vzhľadom na to, že existuje množstvo enginov môže byť výber toho správneho pre váš projekt zložitý. Budem sa však zaoberať iba jedným z najznámejších open source enginov. Tento engine sa volá Unreal Engine a detailne ho popíšem v nasledujúcej kapitole.

1.1 Unreal Engine

Spoločnosť **Epic Games** vyvinula v roku 1998 **Unreal Engine**. Unreal engine je softvérová aplikácia, ktorá bola prvýkrát predstavená v hre Unreal. Aj keď bol pôvodne vyvinutý pre tento účel, svoje uplatnenie si našiel aj v iných žánroch, vrátane platformových hier, simulácií a podobne. Dokonca bol použitý aj na projektoch, ktoré neboli herného typu a to v rôznych odvetviach, ako je automobilový priemysel, stavebníctvo, či strojárstvo. Unreal Engine je napísaný v jazyku C ++, ktorý sa vyznačuje vysokou mierou prenosnosti a podporuje širokú škálu platforiem, ako napríklad: Microsoft Windows, macOS, Linux, iOS, Android, Nintendo Switch, PlayStation 4, Xbox One, ... [3]

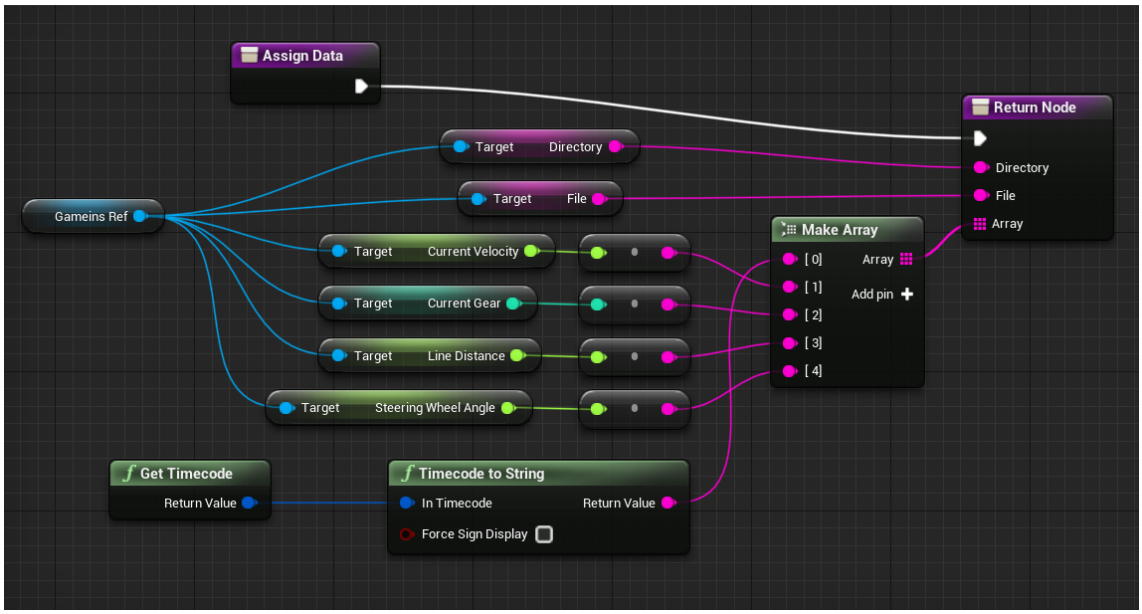
Posledným vydaním je **Unreal Engine 4**, ktorý bol v roku 2014 uvedený na trh, avšak bolo potrebné si ho predplatiť. Od roku 2015 poskytuje Unreal engine open source verziu. V Unreal Engine 4 je možné programovať dvomi spôsobmi a to buď Blueprint Visual Scripting alebo C++. Pri programovaní v C++ je však potrebné mať Visual studio alebo iné IDE (Integrated development environment), pretože má nástroje na kompiláciu C++ kódu. Visual studio je možné otvoriť priamo

z Unreal editoru a to buď vytvorením novej C++ triedy alebo kliknutím už na takto vytvorený Blueprint.

1.1.1 Blueprints Visual Scripting

Je nový herný skriptovací systém v Unreal engine, ktorý je založený na vizuálnom skriptovacom systéme, pri ktorom sa kód vytvára prostredníctvom grafov uzlov a spojení, ktoré vytvárajú skutočný kód. Tento systém v Unreal Engine je mimoriadne výhodný, pretože umožňuje iným používateľom, ako napríklad dizajnérom, vytvárať zložité a prepracované systémy hrania, ktoré boli predtým k dispozícii iba programátorom.[4]

Na Obr. 1.1 môžeme vidieť názornú ukážku kódu naprogramovanú pomocou Blueprint Visual Scripting. Pri zavolaní funkcie Assign Data sa vrátia údaje, ktoré boli uložené do poľa.



Obr. 1.1: Blueprints Visual Scripting

1.1.2 Typy Blueprints

Blueprints môžeme rozdeliť do niekoľkých kategórií podľa ich využitia:

- **Blueprint Class** sa často nazývaná Blueprint, umožňuje tvorcom obsahu ľahko pridať funkcie k existujúcim triedam. Vytvárajú sa v Unreal Editore ako vizuálne prvky. V podstate vytvárajú nový objekt alebo typ herca z už existujúcej triedy, ktorého je potom možné použiť.

- **Data-Only Blueprint** je Blueprint Class, ktorá obsahuje iba premenné a komponenty zdedené od jeho rodiča. Môže ich upraviť alebo vylepšiť. Nemožno však pridávať nové prvky. Taktiež z nich môžeme urobiť úplné Blueprints a to jednoduchým pridaním premennej alebo komponentu pomocou Blueprint editoru.
- **Blueprint Macro Library** je zbierka makier alebo grafov, ktoré je možné použiť ako uzly v iných Blueprintoch. Výhodou je, že môžu šetriť čas, pretože môžu ukladať bežne používané sekvencie uzlov spolu so vstupmi a výstupmi na vykonávanie aj na prenos údajov. Makrá sú zdieľané medzi všetkými grafmi, ktoré na ne odkazujú.
- **Blueprint Utilities** je Blueprint iba pre editor, ktorý je možné použiť na vykonávanie akcií editora alebo na rozšírenie jeho funkcií. Používa sa napríklad pri udalosti(event) bez parametrov, ako sú tlačidlá v používateľskom rozhraní.
- **Blueprint Interface** je zbierka názvov funkcií, bez implementácie, ktoré je možné pridať k iným Blueprintom. V každom Blueprinte, ktorý obsahuje tieto funkcie, je možné pridať konkrétnu implementáciu. Je to v podstate rozhranie, ktoré umožňuje zdieľanie a posielanie rôznych typov objektov a údajov.
- **Level Blueprint** je špeciálny typ Blueprint, ktorý slúži ako globálny graf udalostí na celej úrovni. Každá úroveň v projekte má vytvorený vlastný Level Blueprint. Nové Level Blueprints však nie je možné vytvoriť prostredníctvom editora. Level Blueprint tiež poskytuje kontrolný mechanizmus pre streamovanie úrovne a sekvencer, ako aj pre udalosti aktérov umiestnených na úrovni. [4]

1.1.3 Editor

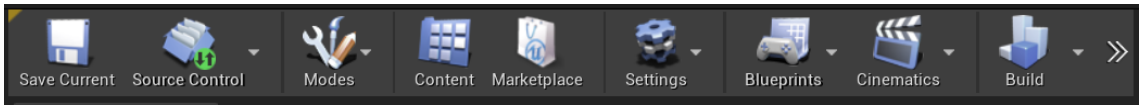
V nasledujúcej podkapitole budem opisovať Unreal Editor, čo je vývojové prostredie pre Unreal Engine. Opíšem iba niektoré základné nástroje, ktoré som používal v tejto práci. V editore sa môžeme nájsť:

- **Tab Bar** sa nachádza v hornej časti editoru a sú to záložky s názvom aktuálnej úrovne. Karty z iných okien editora môžu byť ukotvené vedľa tejto karty pre rýchlu a ľahkú navigáciu, podobne ako vo webovom prehliadači. Názov karty je zároveň názvom úrovne, ktorá sa momentálne upravuje.[6]

Obr. 1.2: Tab Bar[6]

- **Toolbar** Panel s nástrojmi zobrazuje skupinu príkazov, ktoré poskytujú rýchly prístup k bežne používaným nástrojom a operáciám. Napríklad môžeme súbor

ukladať cez Save Current alebo zálohovať cez Source Control. Nachádza sa tam aj Marketplace, cez ktorý sa dá pristúpiť k Unreal Engine Marketplace. Môžeme tu nájsť rôzne komponenty pre UE, ako aj pluginy(napr. VaRest). Cez Settings môžeme zase meniť nastavenia projektu alebo povoliť plugin. Taktiež sa tu dá kompilovať projekt a samozrejme spustiť. Dá sa dokonca vybrať, ako chceme projekt spustiť, či vo viewporte, ďalšom okne a pod.[7]

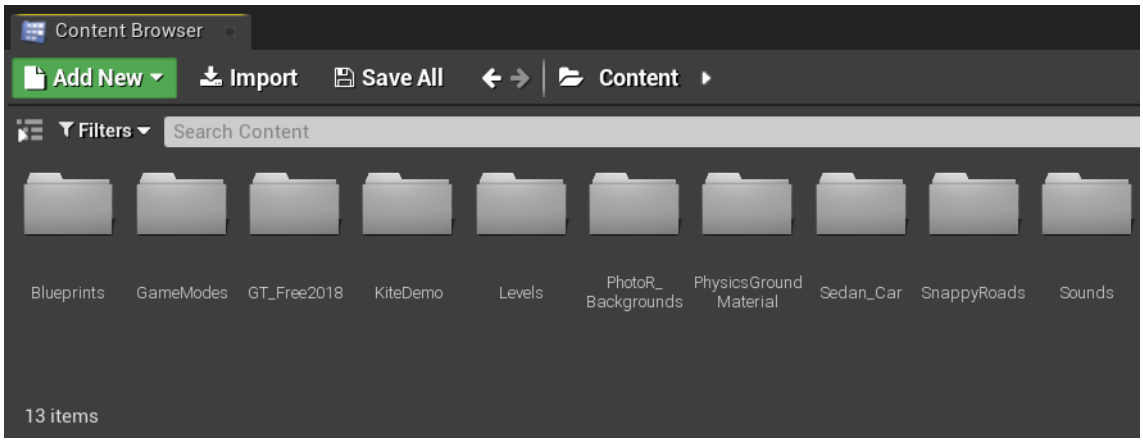


Obr. 1.3: ToolBar

- **Viewport** je hlavná časť editoru, kde je možné upravovať Blueprinty alebo napríklad iné prvky ako widgety a podobne. Nachádza sa v strede editoru. Jednoducho povedané, Viewporty obsahujú rôzne nástroje a vizualizéry. Tie vám pomôžu vidieť presne tie údaje, ktoré potrebujete. Bežnejšie režimy zobrazenia majú svoje vlastné klávesové skratky, ale ku všetkým je prístup z Viewportu cez menu View Mode.[8]

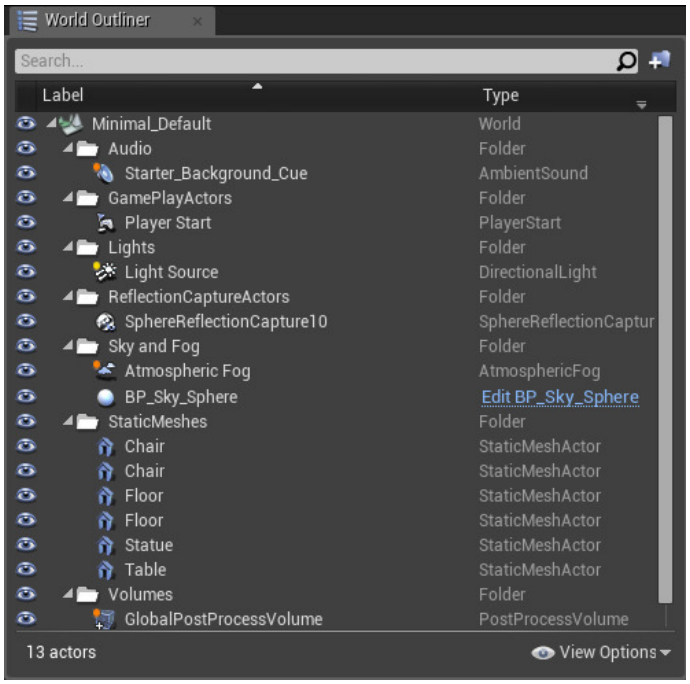
Obr. 1.4: Viewport

- **Content Browser** je v ľavom dolnom rohu. V Content Browser je možné spravovať priečinky s obsahom a vykonávať s nimi ďalšie užitočné operácie, ako napríklad premenovanie, presun, kopírovanie a prezeranie odkazov. Taktiež sa tam nachádza užitočné textové pole (Search Content), ktoré umožnuje rýchle vyhľadávanie a filtrovanie zoznamu.[9]



Obr. 1.5: Content Browser

- **World Outliner** je v pravom hornom rohu a sú v ňom zobrazení všetci herci v scéne v hierarchickom stromovom zobrazení. Hercov je možné vybrať a upraviť priamo z World Outliner. Pomocou rozbaľovacej ponuky Informácie môžete zobraziť ďalší stĺpec, ktorý zobrazuje úrovne, vrstvy alebo názvy ID. Taktiež je tam užitočné textové pole (Search), ktoré umožnuje rýchle vyhľadávanie a filtrovanie zoznamu.[10]



Obr. 1.6: World Outliner[10]

- **Details** -Panel details obsahuje informácie, pomocné programy a funkcie pre aktuálny výber vo Viewporte. Obsahuje editačné polia pre pohyb, rotáciu a zmenu mierky hercov, zobrazuje všetky editovateľné vlastnosti pre vybraných

hercov a poskytuje rýchly prístup k ďalším funkciám úprav v závislosti od typu herca. Vybratí herci môžu byť napríklad prevedení na iný kompatibilný typ.[11]

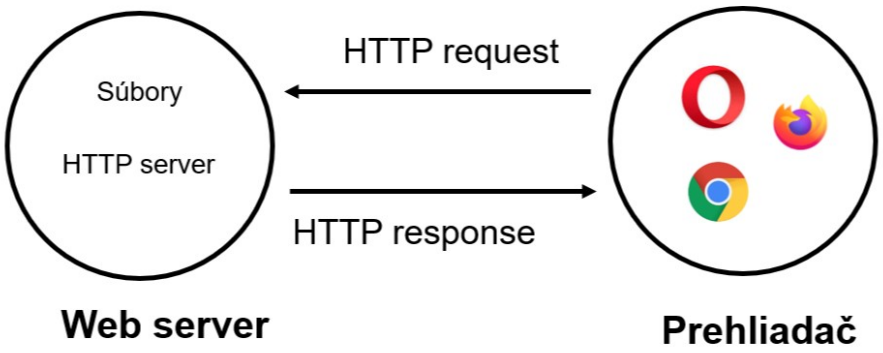
Obr. 1.7: Details[11]

2 Webové servery

Webový server je spojenie softvéru a hardvéru, ktoré dokáže splniť požiadavky od klienta v sieti www (world wide web). Hlavnou úlohou webového servera je zobrazovať obsah webových stránok prostredníctvom ukladania, spracovania a doručovania webových stránok používateľom (klientom). Ak chceme otvoriť nejakú stránku, tak webový prehliadač (klient) pošle požiadavku serveru. Na komunikáciu medzi Klientom a serverom sa používajú takzvané pravidlá komunikácie, ktorým sa hovorí protokoly. Hlavným protokolom je HTTP (Hypertext Transfer Protocol), HTTP cez Secure Sockets Layer (HTTPS), ale používajú sa aj SMTP (Simple Mail Transfer Protocol) a FTP (File Transfer Protocol), ktoré sa používajú na e-mail, prenos súborov a ukladanie.[12]

Doručovanými stránkami sú najčastejšie dokumenty HyperText Markup Language(HTML), ktoré okrem textového obsahu môžu obsahovať aj obrázky, štýly a skripty. Ak pri doručovaní server nedokáže odpovedať obsahom, odpovie chybovou správou.

Zdrojom takýchto súborov je v skutočnosti je úložisko virtuálneho alebo fyzického serveru, ktorý musí zvládnuť odpovedať viacerým klientom naraz.[13]



Obr. 2.1: Komunikácia medzi serverom a prehliadačom

Webový server možno použiť na poskytovanie statického alebo dynamického obsahu. **Statický obsah** je taký, ktorý sa zobrazí raz pri načítavaní stránky. **Dynamický obsah** je možné aktualizovať a meniť. Statický webový server bude pozostávať z počítača a softvéru HTTP servera.

Dynamické webové prehliadače budú pozostávať z webového servera a iného softvéru, ako je napríklad aplikačný server. Považuje sa za dynamické, pretože aplikačný server možno použiť na aktualizáciu akýchkoľvek súborov pred ich odoslaním do prehliadača. Webový server môže generovať obsah, keď je požadovaný z databázy. Tento proces je flexibilnejší, ale aj komplikovanejší.[12]

2.1 Rešerš webových serverov

V nasledujúcej kapitole urobím rešerš webových serveroch. Popíšem niektoré vlastnosti, výhody a nevýhody ich použitia.

2.1.1 Apache HTTP server

Cieľom Apache HTTP server je vyvinúť a udržiavať open-source server HTTP pre moderné operačné systémy, ako je napríklad Windows a UNIX. Tento server by mal poskytovať bezpečný, efektívny a rozšíriteľný server, ktorý ponúka aktuálne štandardy HTTP. Apache patrí medzi najrozšírenejšie webové servery najmä preto, lebo je to open-source server.[14]

Apache je bežne používaný so skriptovacím jazykom PHP a databázou MySQL. Na jednoduchú prácu pre vývojárov s Apache a MySQL bol vytvorený softvérový balíček XAMPP(Cross-Platform, Apache, MySQL a Perl/PHP). Tento softvér veľmi jednoducho vytvorí lokálny webový server pre vývoj a testovanie. Inštalačný balík obsahuje všetko potrebné - serverovú aplikáciu (Apache), databázu (MySQL) a skriptovací jazyk (PHP). XAMPP je multiplatformový, čo znamená, že funguje na Linuxe, na MacOS i Windows.[15]

2.1.2 Internet Information Server

Internet Information Server(IIS) je softvérový balík webového servera určený pre Windows server. Na IIS je možné využívať štandardné webové stránky HTML, ale aj dynamické webové stránky, ako sú napríklad aplikácie ASP.NET a stránky PHP. Keď klient navštívi stránku na statickom webe, služba IIS jednoducho odošle HTML a súvisiace obrázky do prehliadača používateľa. Pri prístupe na stránku na dynamickom webe spustí služba IIS ľubovoľné aplikácie a spracuje skripty obsiahnuté na tejto stránke. Je to populárna voľba pre komerčné webové stránky, pretože ponúka mnoho pokročilých funkcií a je podporovaná spoločnosťou Microsoft. Nevýhodou oproti Apache HTTP je však cena, ktorá rastie v závislosti od počtu používateľov.[16].

2.1.3 Nginx

Je vysoko výkonný open-source HTTP server a reverzným proxy serverom, ako aj proxy server IMAP / POP3. NGINX je známy pre vysoký výkon, stabilitu, bohatú sadu funkcií, jednoduchú konfiguráciu a nízku spotrebu zdrojov. Nginx je dostupný na Unixe, Linuxe a ďalších Unix-like systémoch pod BSD; existujú varianty pre Solaris, MacOS i MS Windows.

NGINX je jedným z mála serverov naprogramovaných na riešenie problému C10K. Na rozdiel od tradičných serverov sa NGINX nespolieha na vlákna pri vybavovaní požiadaviek. Namiesto toho používa oveľa škálovateľnejšiu (asynchrónnu) architektúru riadenú udalosťami. Táto architektúra využíva pri načítaní malé, ale čo je dôležitejšie, predvídateľné množstvo pamäte. Aj keď neočakávate, že vybavíte tisíce simultánnych požiadaviek, stále môžete ťažiť z vysoko výkonného a malého pamäťového priestoru NGINX. NGINX sa rozširuje vo všetkých smeroch: od najmenšieho VPS až po veľké klastre serverov.[17]

2.1.4 Lighttpd

Je open source webový server dodávaný s operačným systémom FreeBSD. Lighttpd je bezpečný, rýchly, kompatibilný a veľmi flexibilný webový server, ktorý bol optimalizovaný pre vysokovýkonné prostredia. V porovnaní s ostatnými webovými servermi má veľmi malú pamäťovú stopu a stará sa o načítanie procesora. Jeho pokročilá sada funkcií (FastCGI, CGI, Auth, výstupná kompresia, prepisovanie adries URL a mnoho ďalších) robí z Lighttpd perfektný softvér webového servera pre každý server, ktorý trpí problémami s načítaním.

Jeho architektúra je založená na udalostiach a optimalizovaná pre veľké množstvo paralelných pripojení (keep-alive), ktoré sú dôležité pre vysoko výkonné aplikácie AJAX.[18]

2.1.5 OpenResty

Je webová platforma, ktorá je vylepšenou verziou Nginx. Nachádzajú sa v nej knižnice Lua, vysoko kvalitné moduly Nginx tretích strán a väčšina externých komponentov. Je navrhnutý tak, aby pomáhal vývojárom ľahko vytvárať škálovateľné webové aplikácie, webové služby a dynamické webové brány.

Vďaka dobre navrhnutým modulom Nginx (z ktorých väčšinu si vyvíja samotný tím OpenResty), OpenResty efektívne premení server Nginx na výkonný server webových aplikácií, v ktorom môžu vývojári webu používať skriptovací jazyk Lua. OpenResty umožňuje konštruovať mimoriadne výkonné webové aplikácie, ktoré sú schopné zvládnuť 10 000 až 1 000 000+ pripojení.[19]

2.2 Porovnanie Apache vs Nginx

Apache a Nginx patria medzi najpoužívanejšie webové servery. Podľa netcraftu [20] sa stal najpoužívanejší Nginx(35% všetkých web stránok) a druhý je Apache(26% web stránok).

Jedným z rozdielov medzi nimi je, že Apache dokáže spracovať dynamický obsah (napr. PHP) a Nginx potrebuje komponenty tretích strán(externé), cez ktoré spracúva dynamický obsah. Funguje to tak, že Nginx pošle tento dynamický obsah komponentu a ten sa spracuje na statický obsah (JSON, HTML a pod). Takto spracovaný obsah sa následne pošle späť Nginx. Tento proces by mal trvať rovnako dlho pri obidvoch serveroch.[21]

Hlavným rozdielom medzi nimi je však rýchlosť, akou dokážu spracovať statické súbory. Podľa niektorých meraní je Nginx až 2,5 krát rýchlejší, a preto je vhodnejšie voliť tento server, ak je viac statického obsahu ako dynamického.[21]

V nasledujúcej tabuľke č.1 som názorne porovnal kľúčové rozdiely medzi serverom Apache HTTP a serverom Nginx, ktoré by mohli ovplyvniť výber medzi týmito servermi.

Tab. č.1: Porovnanie Apache a Nginx

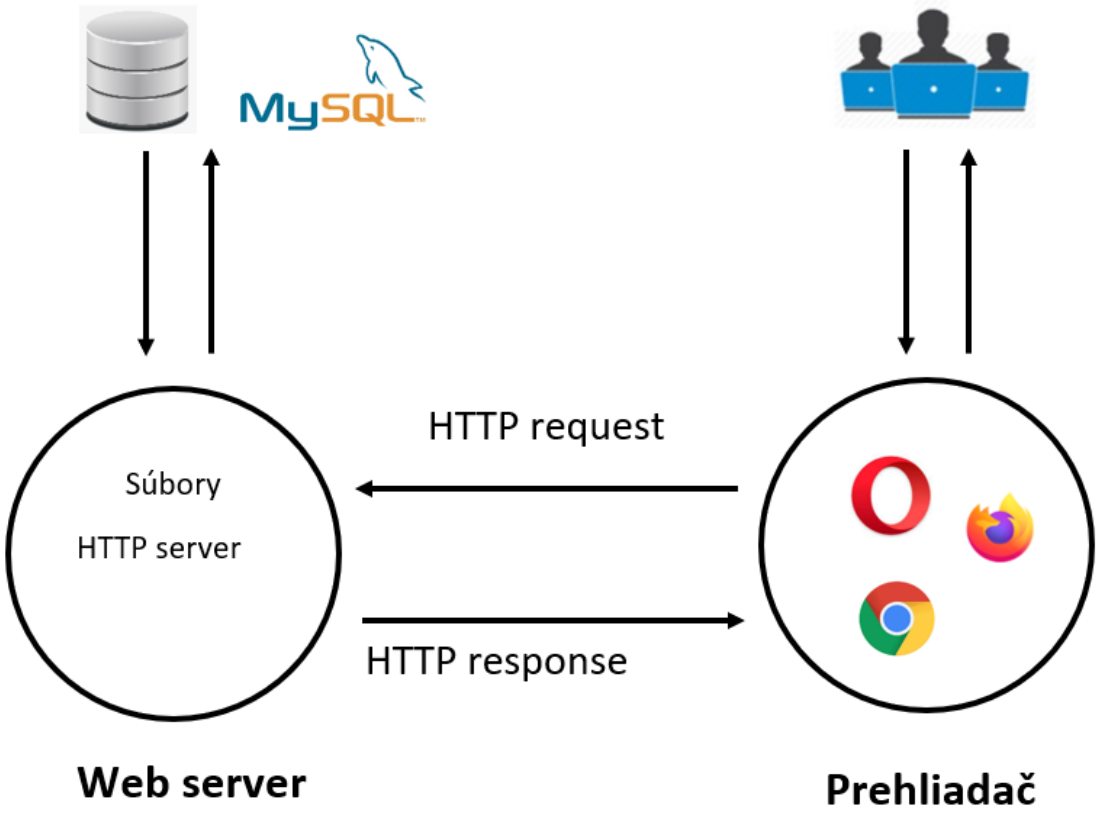
Funkcia	**Apache**	**Nginx**
Podpora OS	podporuje Windows aj Unix(Linux a BSD)	podporuje Unix, ale Windows len čiastočne
Spracovanie požiadaviek	nedokáže spracovať viacero požiadaviek naraz	dokáže spracovať viacero požiadaviek naraz
Dynamický obsah	dokáže spracovať dynamický obsah	nedokáže ho samostatne spracovať
Statický obsah	Zobrazovanie statického obsahu je pomalé	2,5-krát rýchlejší ako Apache a zaberá menej pamäte
Účel	navrhnutý ako webový server	navrhnutý ako proxy server
Dynamické moduly	jednotlivé moduly sú dynamicky nahrávané	moduly sa nedajú nahrávať dynamicky, musia sa najskôr spracovať
Základná Architektúra	má viac vláknovú architektúru, každý nový klient je nové vlákno	asynchrónny prístup riadený udalosťami na zvládnutie viacerých požiadaviek
Bezpečnosť	zabezpečený webový server	zabezpečený webový server
Flexibilita	Je možné ho prispôsobiť pridaním modulov. Apache mal ako prvý dynamické načítanie modulov.	NGINX verzia 1.11.5 a NGINX Plus Release R11 predstavili kompatibilitu pre dynamické moduly.
Podpora a dokumentácia	vynikajúca podpora a dokumentácia	vynikajúca podpora a dokumentácia

3 Databáza

Je množina štruktúrovaných dát, ktoré sú uložené na serveri. Prevažne slúži na ukladanie, triedenie a spätné vyhľadávanie dát. K dátam pristupujeme pomocou programu, ktorý využíva jazyk pre databázy, ako napríklad SQL. Skratka SQL je skrátene Structured Query Language, čo je možné preložiť ako štruktúrovaný vyhľadávací jazyk.

SQL je používaný takmer všetkými relačnými databázami k vytváraniu otázok(napr. INSERT, SEARCH alebo DELETE), definovanie údajov, manipuláciu s dátami a na riadenie prístupu. Jazyk SQL vyvinula v 70. rokoch spoločnosť IBM v spolupráci so spoločnosťou Oracle. Vznikla implementáciou štandardu SQL ANSI. Jazyk SQL získal mnoho rozšírení od spoločností ako napríklad IBM, Oracle a Microsoft. Hoci sa jazyk SQL stále dosť používa, začínajú sa objavovať nové programovacie jazyky[22].

Na nasledujúcom obrázku môžeme vidieť, ako funguje komunikácia s databázou. Keď klient chce dáta z databázy, tak webový prehliadač pošle požiadavok serveru, ktorý ich následne vytiahne a pošle dáta späť.



Obr. 3.1: Komunikácia medzi databázou a užívateľom

3.1 Typy Databáz

Databáz je niekoľko druhov. Odlišujú sa spôsobom použitia. Tu sú niektoré z nich[22]:

- **Relačná databáza** obsahuje dáta sú usporiadané v tabuľkách(ktoré majú medzi sebou vzťahy, väzby) s riadkami a stĺpcami. Je to najflexibilnejší a najefektívnejší spôsob ako pristupovať k súhrnu dát. Medzi najznámejšie relačné databázy patria: MySQL, Maria DB, Microsoft SQL, PostgreSQL, SQlite, Firebird.
- **Objektovo orientovaná databáza** podobne ako v OOP(objektovo orientovanom programovaní) sú dáta v databáze reprezentované objektom.
- **Distribuovaná databáza** sa skladá z dvoch alebo viac súborov nachádzajúcich sa na iných umiestneniach. Môže byť uložená na viacerých počítačoch alebo sieťach.
- **Dátové sklady** sú centrálne úložiská dát, tieto databázy určené pre rýchle zadávanie otázok na analýzu.
- **Databáza NoSQL** je nerelačná databáza, ktorá umožňuje ukladanie a manipuláciu s neštruktúrovanými a čiastočne štruktúrovanými dátami (na rozdiel od relačnej databázy, ktorá definuje formát všetkých dát, ktoré je možné do databázy uložiť). Databáza NoSQL získala popularitu v súvislosti so zvýšeným záujmom o webové aplikácie a ich väčšiu zložitosť. Medzi najznámejšie nerelačné databázy patria: mongoDB, OrientDB, Amazon DynamoDB.
- **Grafová databáza** ukladá dáta vo forme entít a relácií medzi entitami.
- **Databáza OLTP** je rýchla, analytická databáza navrhnutá pre veľký počet transakcií realizovaných viacerými užívateľmi.

3.2 MySQL

je jedna z najpoužívanejších SQL databáz. MySql je vyvíjaná a podporovaná spoločnosťou Oracle Corporation ako open source databáza. Databázový systém je relačný, typu DBMS (database management system). MySQL je podporovaný na viacerých platformách (ako Linux, Windows či Solaris). Väčšinou je tvorená z jednej alebo viac tabuliek, ktoré obsahujú stĺpce a riadky. Je možné nastaviť pravidlá upravujúce vzťahy medzi rôznymi údajovými poľami, ako napríklad primárne a cudzie kľúče medzi rôznymi tabuľkami.[23]

3.3 Maria DB

Je jednou z najpopulárnejších relačných databáz založených na open source systéme. Vytvorili ju pôvodní vývojári MySQL. MariaDB sa používa, pretože je rýchla,

robustná, s bohatým ekosystémom úložných mechanizmov, doplnkov a mnohých ďalších nástrojov. Je veľmi univerzálna pre širokú škálu prípadov použitia.[24]

3.4 PostgreSQL

Je výkonný open source objektovo-relačný databázový systém, ktorý využíva a rozširuje jazyk SQL v kombinácii s mnohými funkciami, ktoré bezpečne ukladajú a rozširujú najkomplikovanejšie dátové úlohy. Databáza si získala silnú reputáciu vďaka svojej osvedčenej architektúre, spoľahlivosti, integrite dát, robustnej množine funkcií, rozšíriteľnosti a odhodlanosti komunity, ktorá stojí za softvérom, aby neustále poskytovala výkonné a inovatívne riešenia. PostgreSQL beží na všetkých hlavných operačných systémoch, je kompatibilný s ACID od roku 2001 a má výkonné doplnky, ako je populárny geopriestorový databázový rozširovač PostGIS. PostgreSQL taktiež umožnuje spojenie s inými dátovými úložiskami ako sú napríklad NoSQL.[25]

3.5 MongoDB

Je dokumentovo orientovaná databáza NoSQL používaná na ukladanie dát s veľkým objemom. Dokumenty sú podobné formátu JSON, ktorý je považovaný za najprirodzenejší spôsob uvažovania o dátach. Taktiež je oveľa expresívnejší a výkonnejší ako tradičný model riadok / stĺpec. MongoDB je open source databáza, ktorá využíva dva typy vzťahov: referenčné a vložené. Umožňuje tiež komunikáciu s MySQL pri použití BI connection. Je to jedna z najlepších bezplatných databáz, ktorá spadá do kategórie NoSQL databáz.[26]

3.6 Amazon DynamoDB

Je nerelačná databáza, ktorá podporuje štruktúry "kľúč - hodnota" a dátové štruktúry dokumentov. Databáza obsahuje zabudované zabezpečenie, zálohovanie a obnovou. ukladá dáta do pamäte cache pre internetové aplikácie. DynamoDB dokáže spracovať viac ako 10 biliónov požiadaviek za deň a okrem toho dokáže podporovať maximá viac ako 20 miliónov požiadaviek za sekundu.[27]

3.7 SQLite

SQLite je knižnica v jazyku C, ktorá implementuje malý, rýchly, samostatný, vysoko spoľahlivý a plne funkčný SQL databázový engine. SQLite je najpoužívanejší databázový engine na svete. SQLite je zabudovaný do všetkých mobilných telefónov a

väčšiny počítačov a dodávaný v nespočetnom množstve ďalších aplikácií, ktoré ľudia používajú každý deň.[28]

Na rozdiel od databáz založených na princípe klient-server, kde je databázový server spustený ako samostatný proces, SQLite číta a zapisuje priamo do bežných diskových súborov. Kompletná databáza SQL s viacerými tabuľkami, indexmi, spúšťačmi a zobrazeniami je obsiahnutá v jednom súbore disku. Formát databázového súboru je multiplatformný - môžete voľne kopírovať databázu medzi 32-bitovými a 64-bitovými systémami alebo medzi architektúrami big-endian a little-endian. Vďaka týmto vlastnostiam je program SQLite obľúbenou voľbou ako formát aplikačných súborov.[29]

4 Simulácia

Simulácia jazdy vozidlom môže slúžiť ako nástroj na vytvorenie situácií z reálneho sveta. Môžeme simulovať rôzne špecifické a zároveň nebezpečné situácie, ku ktorým môže dôjsť počas jazdy autom. Tieto nebezpečné situácie sa dejú každý deň na našich cestách. Pri simulácií však môžeme vytvoriť tieto situácie v prostredí, kde nie je obmedzenie na počet opakovaní alebo obmedzenie v rýchlosti expanzie nameraných dát, kde sú nízke prevádzkové náklady a hlavne je zabezpečené bezpečné prostredie pre vodiča. V mojom prípade ide o simuláciu vytvorenú v Unreal Engine 4 (UE4). Hlavnou výhodou je, že nastavuje hlavné funkcie a prostredie, v ktorom je jednoduché implementovať ovládateľné objekty a scenáre správania. Napríklad je možné namodelovať dráhu pre závodné auto a merať závislosť srdcového tepu na rýchlosti vozidla. Ďalšou možnosťou by mohla byť simulácia v meste. V tejto simulácii by mohli byť rôzne kritické situácie, pri ktorých by sa skúmala reakčná doba vodiča. Tieto informácie by bolo možné ďalej použiť pri vyvíjaní autonómnych vozidiel, ktoré by vedeli, ako sa majú v daných situáciách zachovať.[30]

4.1 Použitý hardvér

Pri simulácii jazdy vozidlom je potrebný dostačujúci hardvér. Celé vybavenie sa nachádza v učebni, kde celá simulácia bude prebiehať. Na Obr. 4.1 je možné vidieť použitý setup, ktorého súčasťou je:

- Procesor: AMD Ryzen 5 2600 šesťjadrový procesor s taktom 3,40 GHz
- Grafická karta: Radeon RX 590
- HDD: SSD Disk Crucial MX500 500GB
- Operačná pamäť: HyperX 16GB DDR4 3200MHz
- Základná doska: MSI B450 TOMAHAWK
- Monitor: 49"Samsung C49HG90
- Volant a pedály: Logitech G920
- Riadiaca páka: Driving Force Shifter od spoločnosti Logitech

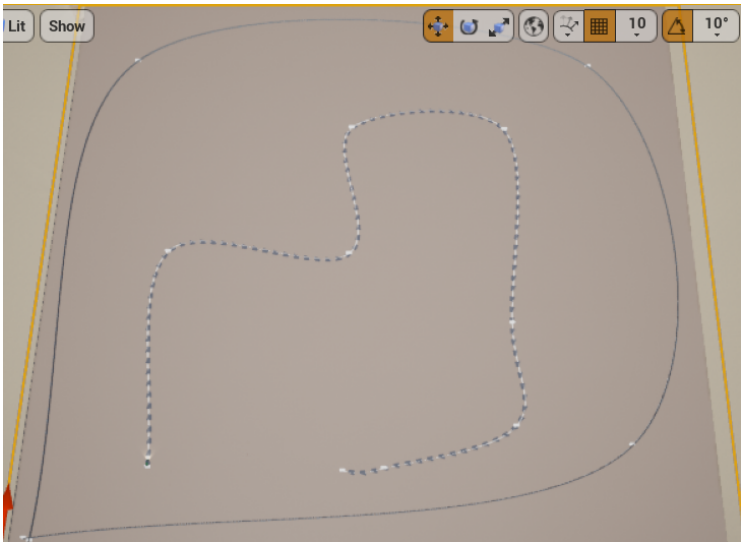
Na tomto hardvéri je možné odsimulovať jazdu autom, pretože obsahuje všetko, čo sa nachádza v aute. Súčasťou je aj rýchlostná páka, pedále a volant.

4.2 Dráha simulácie

V mojej bakalárskej práci som používal scenár, ktorý mi poskytol vedúci bakalárskej práce. Tento scenár bude následne použitý na meranie potrebných dát a ich následné vyhodnocovanie. Na Obr. 4.2 môžete vidieť spomínanú dráhu, po ktorej bude jazdiť auto v simulácii. Dráha slúži ako testovacia, aby sa kalibrovali prístroje a vodič auta

Obr. 4.1: Simulátor v učebni

v simulácii. Dráha pozostáva z rovín a zákrut, pri ktorých sa to dá overiť. Namerané dáta budem ukladať do súboru a aj do databázy na serveri. Ako tieto dáta budem ukladať, ukážem v nasledujúcej kapitole.



Obr. 4.2: Dráha

5 Ukladanie dát

V tejto kapitole popíšem, ako ukladám dáta zo simulácií. Najskôr popíšem ukladanie do súboru a potom ukladanie do databázy. Popíšem môj postup ukladania dát a aký typ formátu dát som zvolil.

5.1 Ukladanie do súboru

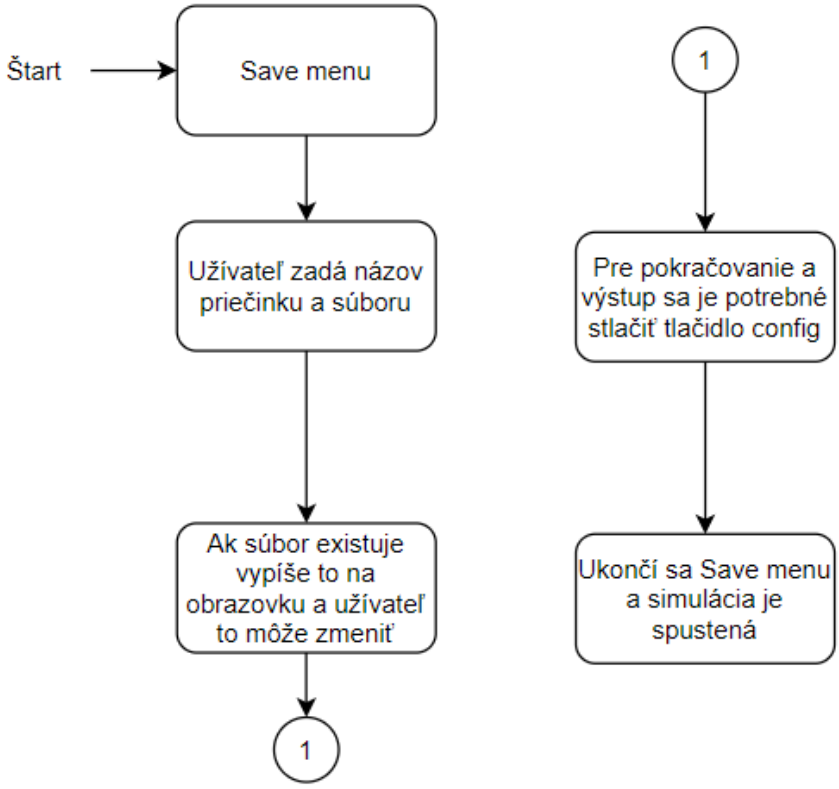
Pri ukladaní do súboru som zvolil postup, pri ktorom sa užívateľovi pri spustení simulácie zobrazí Save menu. V Save menu si zadá názov priečinku a názov súboru, do ktorého sa mu uložia dáta. Následne sa budú ukladať dáta počas celej simulácie vo formáte CSV.

5.1.1 Save menu

Pri spustení aplikácie sa najskôr zobrazí Save menu widget, v ktorom si môžeme určiť názov priečinku a súboru. Do tohoto súboru sa nám budú ukladať všetky dáta zo simulácie jazdy vozidla. Výzor Save menu je dôležitý, lebo je to to prvé, čo uvidíme. Zvolil obrázok priamo zo simulácie, aby som odkazoval priamo na to, čo sa nej bude diať. Pri zadávaní mena súboru alebo priečinku sa taktiež zobrazuje nápoveda, ktorá má pomôcť k jednoduchšiemu používaniu, alebo naviesť užívateľa k tomu, čo má urobiť. Na Obr. 5.1 je ukážka popisovaného Save menu.

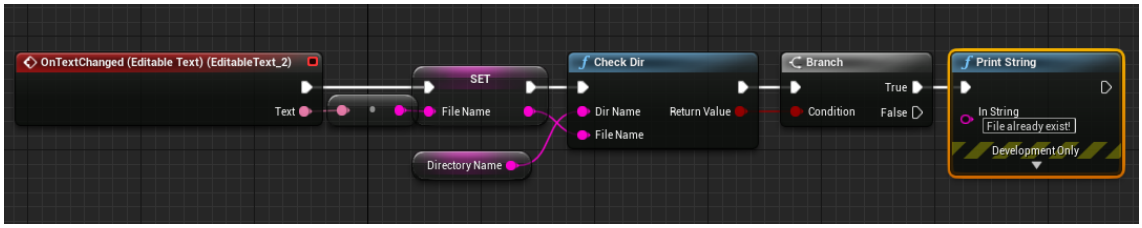
Obr. 5.1: Save menu

Pre jednoduchú predstavu sledu udalostí v save menu a jeho lepšie pochopenie, som vytvoril nasledujúci vývojový diagram. Tento vývojový diagram je označený, ako obrázok č. 5.2. Môžeme na ňom vidieť kroky programu od spustenia save menu až po jeho ukončenie.



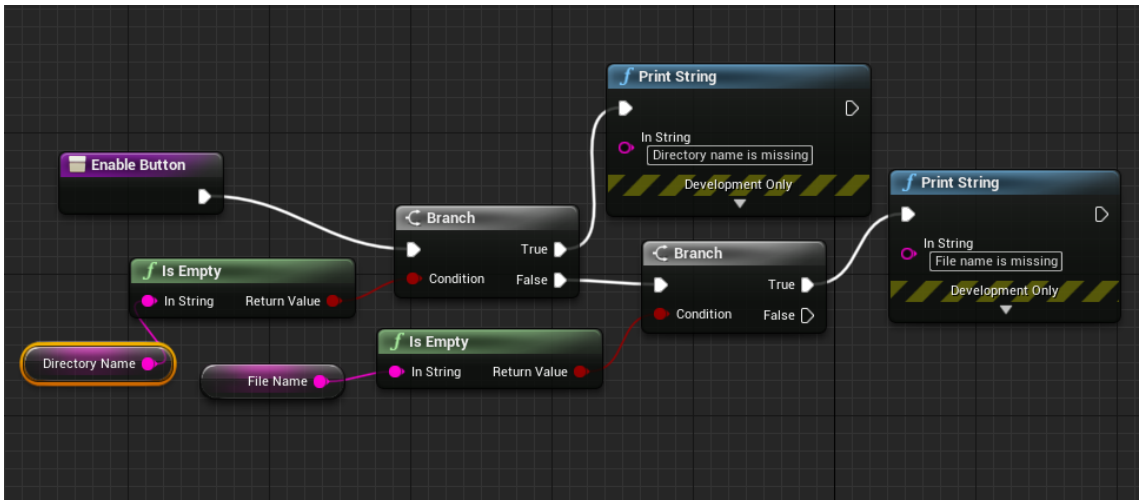
Obr. 5.2: Vývojový diagram

Ak po spustení save menu užívateľ zadá meno priečinku a súboru a tento súbor už existuje, vypíše sa na displej "File already exist" a buď uživateľ zmení názov súboru alebo dáta sa budú zapisovať na koniec už vytvoreného súboru. Túto funkcionalitu zabezpečí kód, ktorý je na obrázku 5.3.



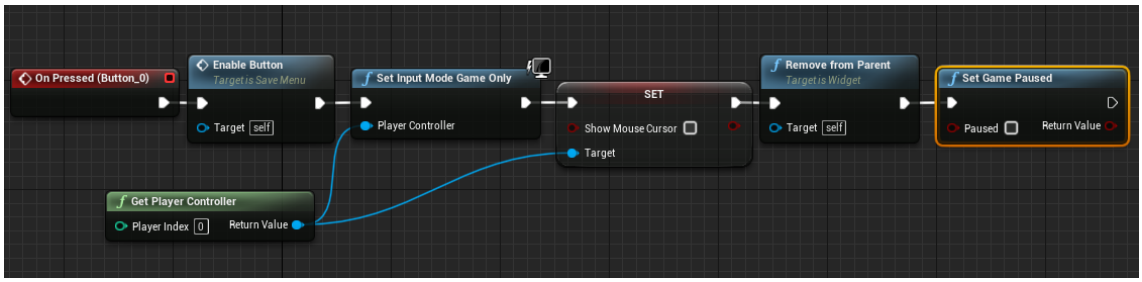
Obr. 5.3: Kontrola, či súbor existuje

Ak chce užívateľ pokračovať ďalej, musí stačiť tlačidlo Confirm (potvrdiť). Toto tlačidlo je možné stlačiť iba v prípade, že názov súboru (a aj priečinku) nie je zhodný s prázdnym stringom, teda názvy neboli zadané. Ak užívateľ stlačí toto tlačidlo a nebude vyplnený názov priečinku, tak sa na obrazovku vypíše hláška "Directory name is missing". To isté, len s obmenou "File name is missing" sa zobrazí, keď bude chýbať názov súboru. Na obrázku 5.4 môžete vidieť ukážku kódu, ktorý zabezpečuje túto funkcionalitu.



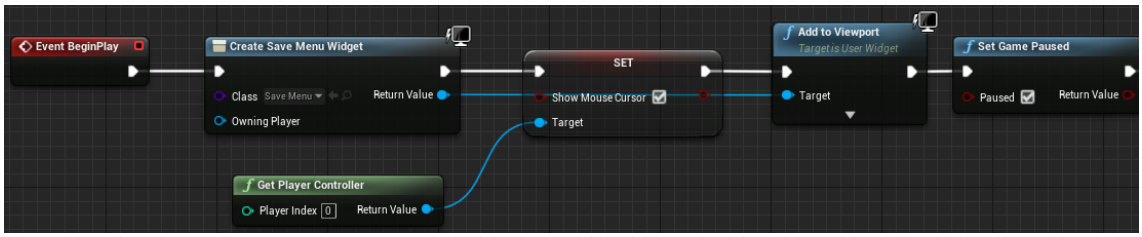
Obr. 5.4: Kontrola zadaných údajov

Keď však stlačí tlačidlo a podmienky budú splnené, tak sa ukončí save menu, odstráni sa z viewportu a hra sa spustí. Taktiež sa hra prepne z pause modu, v ktorom bola. Z dôvodu aby sme obmedzili dáta, ktoré by boli zbytočné, pretože sme simuláciu reálne ešte nezačali. Časť kódu, ktorý toto zabezpečuje, môžete vidieť na obrázku 5.5.



Obr. 5.5: Tlačidlo confirm

Na Obr. 5.6 môžete vidieť ukážku volania widget Save menu. Tento widget je volaný hneď pri začiatku simulácie (pridaný do viewportu a opustí ho až pri stlačení tlačidla confirm.



Obr. 5.6: Save menu widget

5.1.2 Dátový formát

Pri ukladaní do súboru som zvolil ukladanie vo formáte **Comma-separeted values** skrátene CSV. CSV je jednoduchý súbor vo formáte obyčajného textu, určený na ukladanie tabuľkových dát.Dáta v súbore sú oddelené bodkočiarkou. Tieto súbory sa často používajú na výmenu údajov medzi rôznymi aplikáciami. Príklad textového súboru by bol napríklad:

> Time; Velocity; Gear; Line Distance; Wheel Rotation;
> 09:52:24:10;6.0;1;0.058118;0.0; 09:52:24:12;6.0;1;0.056956;0.0;
> 09:52:24:17;7.0;1;0.052962;-1.0; 09:52:24:19;7.0;1;0.05309;-1.0;
> 09:52:24:21;8.0;1;0.055063;-1.0; 09:52:24:22;8.0;1;0.059137;-1.0;
> 09:52:25:00;8.0;1;0.065516;0.0; 09:52:25:02;9.0;1;0.0696;0.0;
> 09:52:25:04;9.0;1;0.070265;0.0; 09:52:25:06;9.0;1;0.070187;1.0;

Na Obr. 5.7 môžete vidieť dáta zo súboru zobrazené v Microsoft Excel.

	A	B	C	D	E
1	Time	Velocity	Gear	Line Distance	Wheel
27	09:52:24:10	6.0	1	0.058118	0.0
28	09:52:24:12	6.0	1	0.056956	0.0
29	09:52:24:14	7.0	1	0.055706	0.0
30	09:52:24:15	7.0	1	0.054378	0.0
31	09:52:24:17	7.0	1	0.052962	-1.0
32	09:52:24:19	7.0	1	0.05309	-1.0

Obr. 5.7: CSV súbor zobrazený v Microsoft Excel

Ukladanie do súboru na lokálnom disku má viacero výhod, napríklad v tom, že je možne pristupovať k dátam aj offline. Dalšiou výhodou môže byť, že nie je potrebné vedieť narábať s databázami (pri malom objeme dát, čo môže uľahčiť prácu). Veľkou nevýhodou je, že nie je možné pristupovať k dátam zo všadiaľ a kedykoľvek, ako pri variante s databázou. Najlepšie by bolo mať obidve riešenia.

5.1.3 Vytvorenie funkcie na ukladanie

Na ukladanie dát do súbora neexistuje žiadena vstavaná funkcia od UE4, a preto som musel vytvoriť svoju vlastnú. Vytvoril som funkciu CheckDir a SaveData. Tieto funkcie môžete nájsť v prílohách ako Výpis A.1. Jej hlavičkový súbor je možné vidieť na Výpis 5.1. Aby som mohol ukladať dáta do súboru, musel som vytvoriť novú Blueprint Function Library.

Blueprint Function Libraries slúži ako zbierka statických funkcií, ktoré nie sú viazané na konkrétny objekt hry. Tieto knižnice možno zoskupiť do množín logických funkcií, napr. AI Blueprint Library, alebo obsahujú úžitkové funkcie, ktoré poskytujú prístup k mnohým rôznym funkčným oblastiam, napr. System Blueprint Library. Blueprint Function Libraries sú veľmi podobné funkciám Blueprint pomocou makra UFUNCTION (). Namiesto odvodenia priamo z UObjectu, dedia všetky knižnice Blueprintu z UBlueprintFunctionLibrary. Mali by obsahovať iba statické metódy[5].

Výpis 5.1: Ukážka hlavičkového súboru SaveFileFL.h

```
1  #pragma once
2
3  #include "CoreMinimal.h"
4  #include "Kismet/BlueprintFunctionLibrary.h"
5  #include "SaveFileFL.generated.h"
6
7  UCLASS()
8  class BP_VISU_PROJECT_API USaveFileFL : public
9  UBlueprintFunctionLibrary
10 {
11   GENERATED_BODY()
12
13     UFUNCTION(BlueprintCallable, Category = "SaveFile",
14     meta = (Keywords = "Save"))
15     static bool SaveData(FString Directory, FString
16     FileName, TArray<FString> Data);
17
18     UFUNCTION(BlueprintCallable, Category = "SaveFile")
19     static bool CheckDir(FString DirName, FString
20     FileName);
21 };
```

Vo výpis A.1 môžete vidieť ukážku zdrojového kódu v C++. Vytvoril som funkciu CheckDir, ktorá má dva vstupné parametre a overuje, ktoré sú zadávané v

Save Menu. Funkcia overí, či súbor s týmto názvom existuje a ak áno, vráti hodnotu true. Funkcia SaveData, ktorá overí, či súbor existuje a ak nie, tak doň zapíše hodnoty "Time; Velocity; Gear; Line Distance; Wheel Rotation;" a potom zapíše všetky hodnoty, ktoré sú dané vstupným parametrom Data.

5.2 Ukladanie do databázy

Pri ukladaní do databázy bolo potrebné si zvoliť správny webový server a databázu. V kapitole 2 som sa zaoberal webovými servermi. Ako najvhodnejší pre moje použitie sa mi javil Apache HTTP server. Medzi hlavne dôvody výberu Apache patria: podporuje operačný systém Windows, dokáže spracovať dynamický obsah a je voľne dostupný. Databázy som zase opisoval v kapitole 3. Pri svojom projekte som zvolil databázu MySQL. MySQL je relačná databáza, je zadarmo a dá sa s ňou jednoducho s ňou pracovať. Na prácu s touto kombináciou som si nainštaloval softvérový balík XAMPP, ktorý som spomínal v kapitole 2.1 pri Apache serveri.

Namiesto PHP som používal Laravel, čo je framework pre PHP. Laravel je voľne dostupný a uľahčuje prácu s PHP. Využíva softvérovú architektúru MVC, čo je skratka pre model-view-controler architektúru. Model obsahuje aplikačné dáta a funkcie, View slúži k prezentácii dát napríklad v HTML, Controller spravuje interakcie medzi užívateľom modelom a pohľadom (view).

Dáta z UE4 som ukladal vo formáte JSON, čo je skratka pre JavaScript Object Notation. JSON je ľahký formát na výmenu údajov. Pre ľudí je ľahké čítať a písať. Stroje môžu ľahko analyzovať a generovať. JSON je textový formát, ktorý je úplne nezávislý od jazyka, ale využíva konvencie známe programátorom jazykov C, vrátane C, C ++, C #, Java, JavaScript, Perl, Python a mnoho ďalších. Tieto vlastnosti robia z JSON ideálny jazyk na výmenu údajov.[31]

```
{
    "simulation_id": 1,
    "time": "2021-05-09 12:34:32",
    "velocity": 13,
    "gear": 2,
    "updated_at": "2021-05-09 12:34:32",
    "created_at": "2021-05-09 12:34:32"
}
```

Obr. 5.8: Ukážka formátu JSON

Tento formát údajov som zvolil, lebo je vhodný na ukladanie do databáz. Ako môžeme vidieť na obrázku 5.8, prvý je ako prvý zadaný cudzí kľúč, aby boli dáta

zaradené do správnej tabuľky.Ďalej nasledujú samotné dáta zo simulácie.

5.2.1 Návrh databázy

Pri vytváraní dataázy bolo potrebné zorganizovať dáta do tabuliek, aby boli prehľadné a mohol som k nim jednoducho pristupovať pri vytváraní webovej aplikácie.

Vytvoril som 3 tabuľky, do ktorých sa budú ukladať dáta. Prvá je tabuľka Drivers(vodiči), v ktorej sa ukladajú dáta o vodičoch. Ukladá sa ich meno, vek a kedy boli vytvorení. Keby došlo k situácii, že sa objavia vodiči s rovnakým menom, môžeme ich stále rozlíšiť cez ID a dátumom kedy boli vytvorení. Príklad tejto tabuľky môzete vidieť na obrázku 5.9.

id	first_name	last_name	age	created_at	updated_at
1	Dominik	Viater	23	2021-05-04 09:48:30	2021-05-04 09:48:30
2	Lubko	Voska	23	2021-05-04 22:39:30	2021-05-04 22:39:30
3	Vendo	Holubcik	26	2021-05-04 22:39:32	2021-05-04 22:39:32

Obr. 5.9: Tabuľka Drivers

Druhá je tabuľka Simulations. Táto tabuľka obsahuje dáta o všetkých simuláciách. Nachádza sa v nej napríklad sim_date, čo je dátum vytvorenia simulácie(spustenie simulácie v UE4). Tiež obsahuje aj driver_id, čo je cudzí kľúč pre tabuľku Drivers. Ukážka tabuľky je na obrázku č. 5.10.

id	driver_id	sim_date ▾ 1	created_at	updated_at
29	1	2021-05-19 10:25:04	2021-05-19 08:25:04	2021-05-19 08:25:04
28	1	2021-05-18 21:31:54	2021-05-18 19:31:55	2021-05-18 19:31:55
27	1	2021-05-06 11:34:34	2021-05-06 09:34:34	2021-05-06 09:34:34
26	1	2021-05-05 21:47:56	2021-05-05 19:47:57	2021-05-05 19:47:57

Obr. 5.10: Tabuľka Simulations

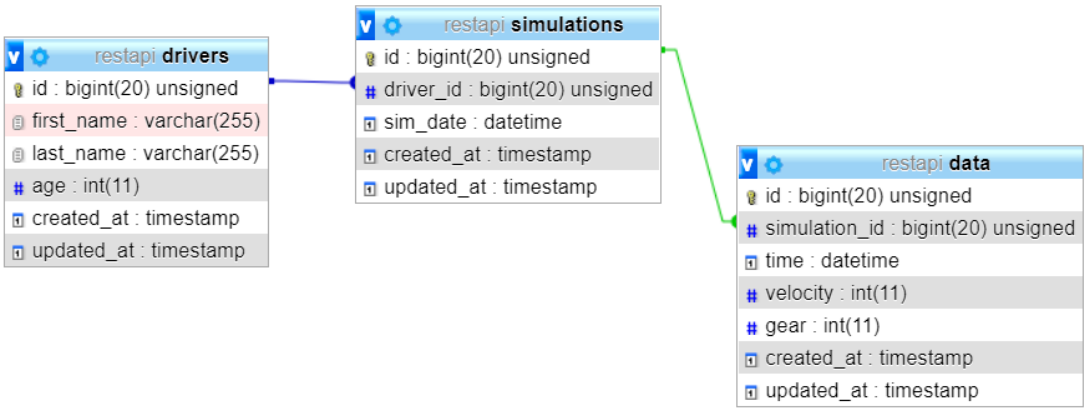
Tretia je tabuľka Data. V tejto tabuľke sa nachádzajú už konkrétne dáta zo simulácie jazdy vozidlom. Ako môžeme vidieť na obrázku 5.11, nachádza sa tam napríklad velocity(rýchlosť) alebo gear(zaradený stupeň). Je tam aj simulation_id, čo je cudzí kľúč pre tabuľku Simulations.

id	simulation_id	time	velocity	gear	created_at	updated_at
1	1	2021-05-04 12:53:32	0	1	2021-05-04 12:53:32	2021-05-04 12:53:32
2	1	2021-05-04 12:59:32	0	1	2021-05-04 11:02:22	2021-05-04 11:02:22
3	1	2021-05-04 12:59:32	1	1	2021-05-04 20:19:45	2021-05-04 20:19:45
4	1	2021-05-04 12:59:32	2	1	2021-05-04 20:19:52	2021-05-04 20:19:52
5	1	2021-05-04 12:59:32	3	1	2021-05-04 20:19:58	2021-05-04 20:19:58
6	1	2021-05-04 12:59:32	4	1	2021-05-04 20:20:03	2021-05-04 20:20:03

Obr. 5.11: Tabuľka Data

K týmto dátam som pristupoval cez phpMyAdmin. Je to bezplatný softvérový nástroj napísaný v PHP, určený na správu MySQL cez web. phpMyAdmin podporuje širokú škálu operácií na MySQL a MariaDB. Často používané operácie (správa databáz, tabuliek, stĺpcov, vzťahov, indexov, používateľov, povolení atď.) je možné vykonávať prostredníctvom používateľského rozhrania, pričom stále máte možnosť priamo vykonávať akýkoľvek príkaz SQL.[32]

Relácie medzi týmito tabuľkami je možné zobrazil aj graficky. PhpMyAdmin ponúka funkciu dizajnér, v ktorej môžeme vidieť všetky vzťahy medzi všetkými tabuľkami v databáze. Na obrázku 5.12 je vidieť spomínané relácie.



Obr. 5.12: Vzťahy medzi tabuľkami

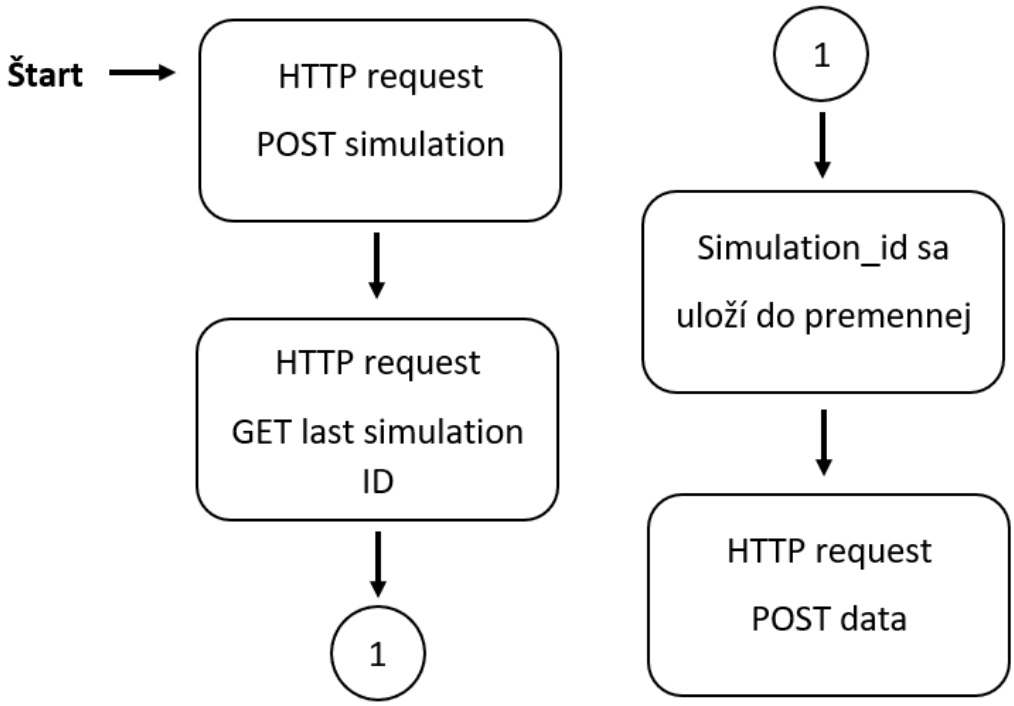
5.2.2 Ukladanie v UE4

Aby bolo možné ukladať dáta do databázy v UE4, musel som vymyslieť systém , ktorý by zabezpečoval komunikáciu medzi serverom, databázou a UE4. VaRest je

plugin, ktorý to umožňuje a ešte k tomu má plnú podporou dotazov JSON. Niektoré z kľúčových vlastností:

- Flexibilná správa požiadaviek Http / Https s podporou rôznych slovies a typov obsahu
- Nie je potrebné žiadne písanie kódu v C ++, všetko je možné spravovať pomocou Blueprintov
- Blueprintable Json wrapper s takmer plnou podporou funkcií pre Json: rôzne typy hodnôt, polia, atď.

Pri spustení simulácie v UE4 sa vytvorí nová simulácia v tabuľke simulations. Na čo je využitý POST request. Následne cez GET request získam ID poslednej simulácie, čo je cudzí kľúč pre tabuľku Data. Potom sa začnú ukladať dáta do tabuľky Data. Tento postup programu môžete vidieť na nasledujúcom vývojovom diagrame(obrázok č. 5.13)



Obr. 5.13: Vývojový diagram

5.3 Porovnanie

V nasledujúcej kapitole porovnám uložené dáta zo súboru a simulácie. Ako som spomínal v kapitole 5.1.2, dátový formát pri ukladaní do súboru je CSV. Ukladanie

do databázy rozobraté v kapitole 5.2, kde dáta budem ukladať v formáte JSON. Aby sa však tieto údaje čo najjednoduchšie porovnali, tak som dáta z databáza stiahol v formáte CSV. Urobil som tak cez phpMyAdmin, ktorý ponúka funkciu Export. Na obrázku č. 5.14 môžete vidieť dáta, ktoré boli ukladané zo súboru.

	A	B	C	D	E
1	Time	Velocity	Gear	Line Distance	Wheel Rotation
2	20:46:25:17	0.0	0	0.066886	0.0
3	20:46:25:22	0.0	0	0.066886	0.0
4	20:46:25:23	0.0	1	0.066896	0.0
5	20:46:26:04	0.0	1	0.066974	0.0
6	20:46:26:05	0.0	1	0.067121	0.0
7	20:46:26:07	0.0	1	0.067033	0.0
8	20:46:26:08	0.0	1	0.067131	0.0
9	20:46:26:10	0.0	1	0.066975	0.0
10	20:46:26:12	0.0	1	0.066935	0.0
11	20:46:26:13	0.0	1	0.066711	0.0
12	20:46:26:14	0.0	1	0.066701	0.0
13	20:46:26:16	1.0	1	0.066535	0.0
14	20:46:26:17	1.0	1	0.066379	0.0
15	20:46:26:19	1.0	1	0.066164	0.0
16	20:46:26:20	2.0	1	0.065783	0.0
17	20:46:26:22	2.0	1	0.065373	0.0
18	20:46:26:23	3.0	1	0.064865	0.0
19	20:46:27:01	3.0	1	0.06427	0.0
20	20:46:27:02	4.0	1	0.063606	0.0
21	20:46:27:03	4.0	1	0.062854	0.0
22	20:46:27:05	5.0	1	0.062034	0.0
23	20:46:27:06	5.0	1	0.061126	0.0
24	20:46:27:07	5.0	1	0.060149	0.0
25	20:46:27:09	6.0	1	0.059094	0.0
26	20:46:27:10	6.0	1	0.057962	0.0
27	20:46:27:12	6.0	1	0.056751	0.0
28	20:46:27:13	7.0	1	0.055462	0.0

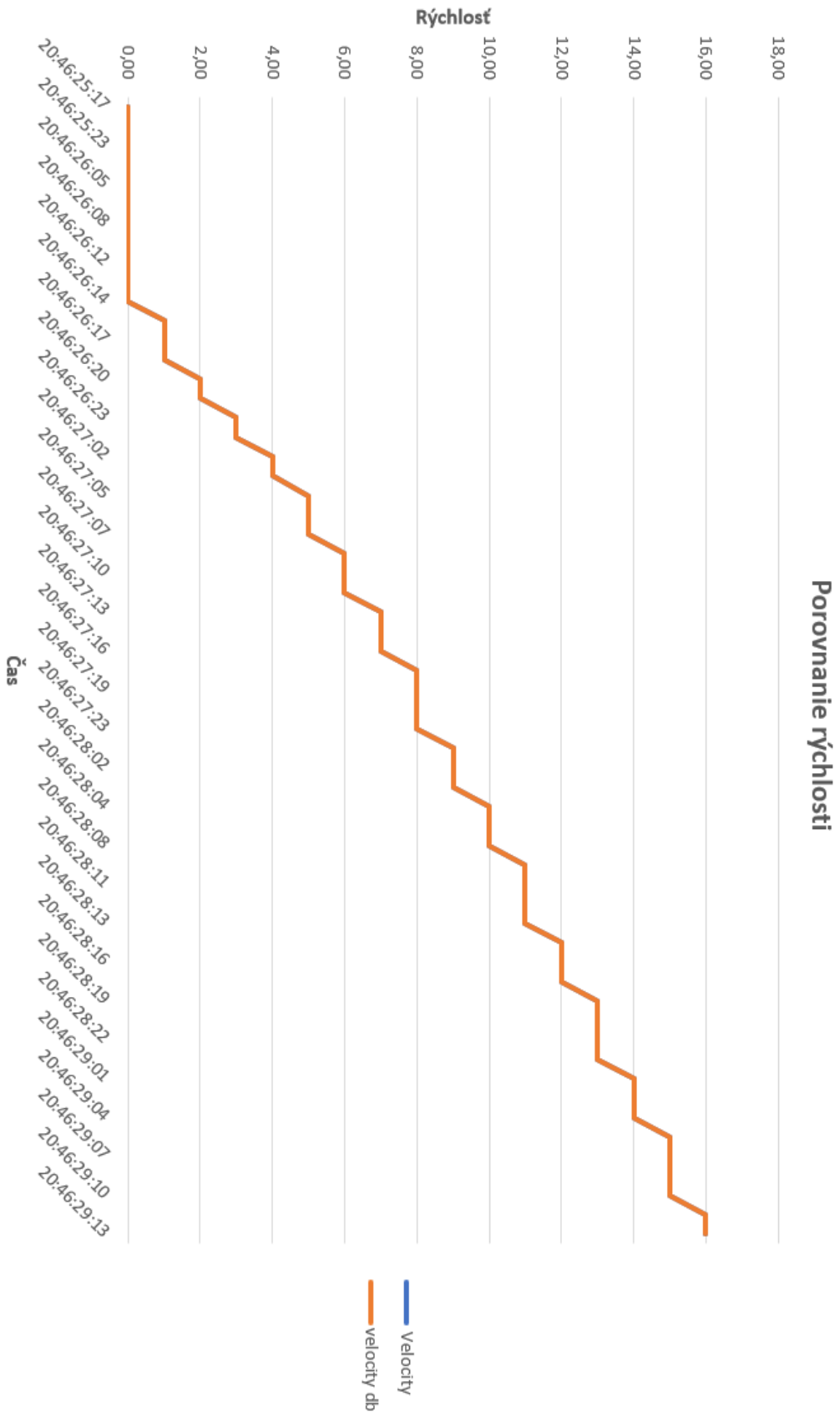
Obr. 5.14: Dáta zo súboru

Na nasledujúcom obrázku č. 5.15 zase sú zobrazené dáta uložené do databázy. Ako si môžete všimnúť napríklad čas a rýchlosť sú zhodné z dátami uloženými do súboru.

	A	B	C	D	E
1	id	simulation_	time	velocity	gear
2	567	30	8:46:25 PM	0	0
3	568	30	8:46:25 PM	0	0
4	569	30	8:46:25 PM	0	1
5	570	30	8:46:26 PM	0	1
6	571	30	8:46:26 PM	0	1
7	572	30	8:46:26 PM	0	1
8	573	30	8:46:26 PM	0	1
9	574	30	8:46:26 PM	0	1
10	575	30	8:46:26 PM	0	1
11	576	30	8:46:26 PM	0	1
12	577	30	8:46:26 PM	0	1
13	578	30	8:46:26 PM	1	1
14	579	30	8:46:26 PM	1	1
15	580	30	8:46:26 PM	1	1
16	581	30	8:46:26 PM	2	1
17	582	30	8:46:26 PM	2	1
18	583	30	8:46:26 PM	3	1
19	584	30	8:46:27 PM	3	1
20	585	30	8:46:27 PM	4	1
21	586	30	8:46:27 PM	4	1
22	587	30	8:46:27 PM	5	1
23	588	30	8:46:27 PM	5	1
24	589	30	8:46:27 PM	5	1
25	590	30	8:46:27 PM	6	1
26	591	30	8:46:27 PM	6	1
27	592	30	8:46:27 PM	6	1
28	593	30	8:46:27 PM	7	1

Obr. 5.15: Dáta z databázy

Pre lepšie vizuálne porovnanie som vytvoril aj graf, kde tieto dáta sú zobrazené v jednom grafe. Toto porovnanie možete vidieť na obrázku č. 5.16.



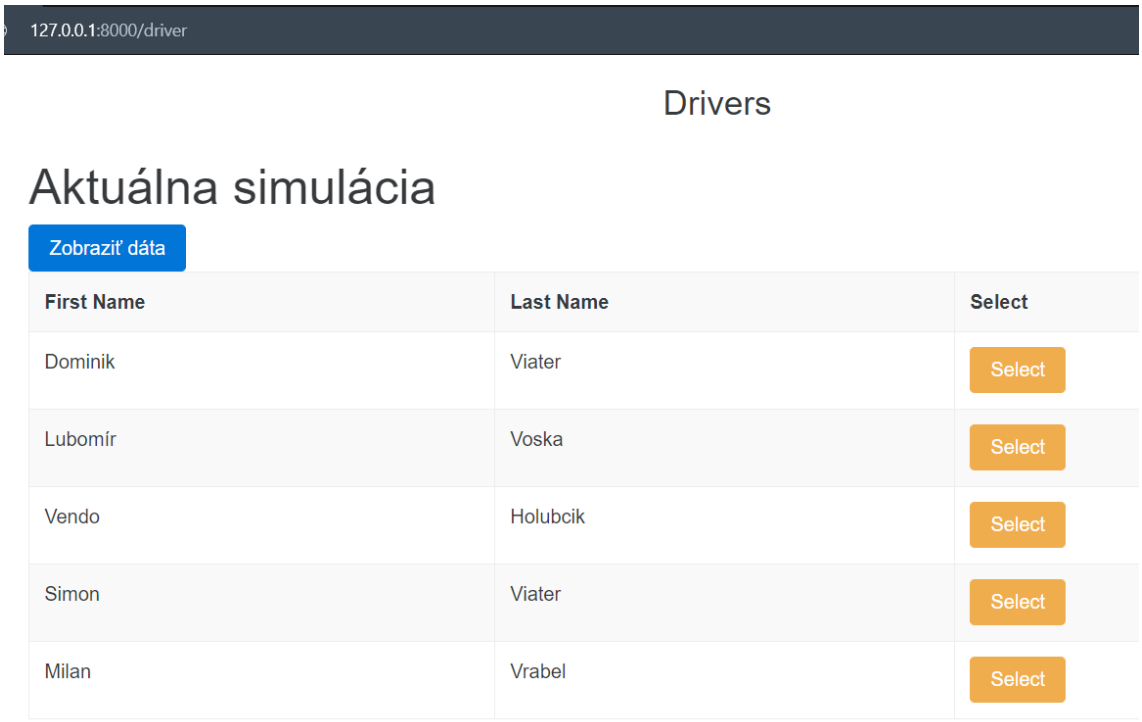
Obr. 5.16: Dáta z databázy

6 Vizualizácia dát

V tejto kapitole detailne popíšem môj vlastný spôsob vizualizácie dát. Pri vizualizácii dát som sa snažil vytvoriť statické aj dynamické stránky. Tieto stránky mali rôzny účel, ktorý popíšem v nasledujúcej kapitole.

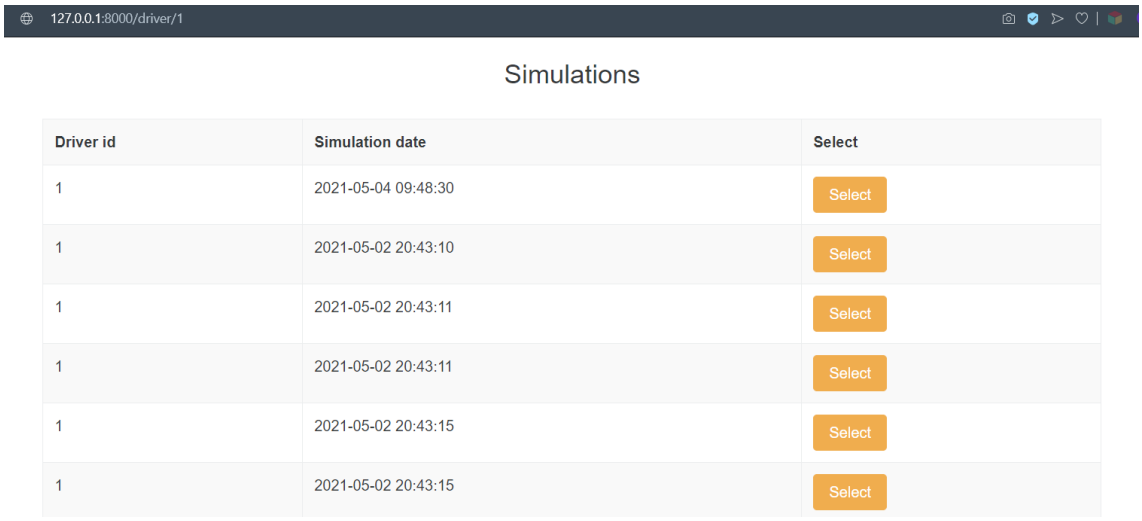
6.1 Statické stránky

Týmito web stránkami som chcel vytvoriť náhradu za sťahovanie do súboru. Chcel som, aby bolo možné po od simulovaní zobraziť všetky dáta pre analýzu. Podarilo sa mi urobiť statické web stránky, na ktorých zobrazujem dáta v tabuľkách viď obrázkok č. 6.1. Zabezpečuje to blade, ktorého ukážku môžete nájsť v prílohách ako výpis B.2. Na tejto web stránke je výpis všetkých vodičov, ktorý simulovali jazdu vozidlom v učebni. Sú zoradení podľa dátumu vytvorenia. Keby sa zaregistrovali boli dvaja s rovnakým menom, mohli by sme ich podľa tohto údaja rozlíšiť.



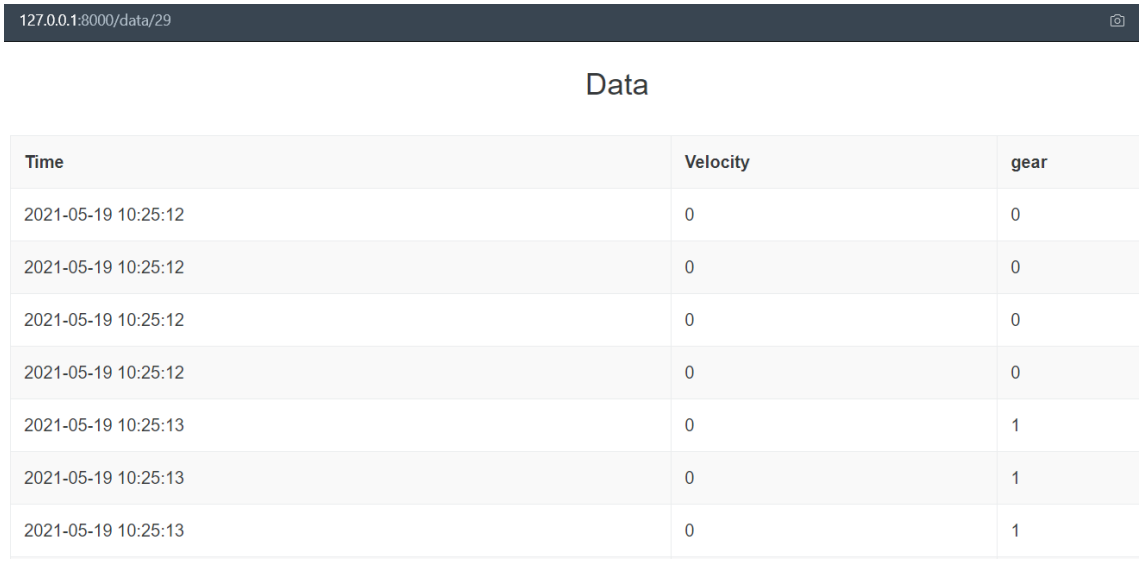
Obr. 6.1: Web stránka Driver

Táto web stránka funguje tak, že si zvolíte vodiča, ktorého simuláciu chcete (tlačidlo Select). Následne sa otvorí stránka so všetkými simuláciami daného vodiča. Tieto simulácie sú zoradené podľa dátumu vytvorenia, teda začatia simulácie. Túto web stránku môžete vidieť na obrázku č. 6.2.



Obr. 6.2: Web stránka Simulation

Následne by ste si vybrali simuláciu (cez tlačidlo Select), z ktorej potrebujete dáta. Tieto dáta sa následne zobrazia na nasledujúcej web stránke. Zobrazia sa v prehľadnej tabuľke. Ukážku môžete vidieť na obrázku č. 6.3.

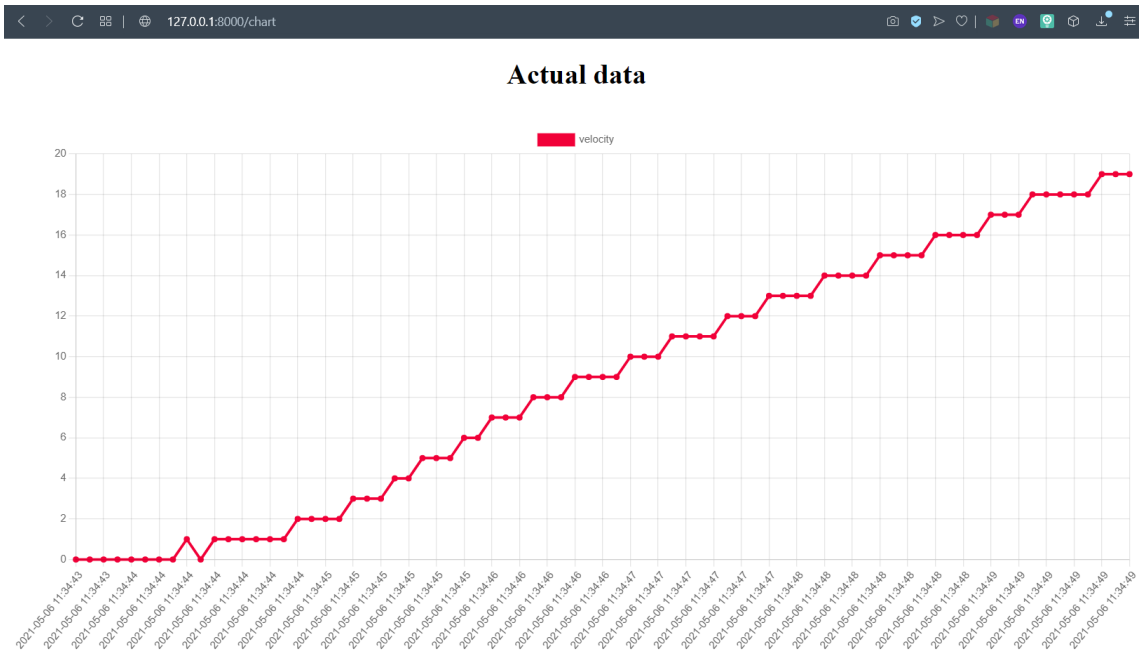


Obr. 6.3: Web stránka Data

6.2 Dynamické stránky

Na zobrazenie dynamického obsahu som vytvoril osobitnú stránku. Na túto web stránku je možné sa dostať cez stránku Driver, o ktorej som hovoril v kapitole č. 6.1. Obsahuje tlačidlo Zobraziť dáta, ktorým zobrazíte dynamický obsah. Na obrázku č.

6.4 môžete vidieť graf, ktorý dáta zobrazuje dynamicky. Je napísaný v Java Script a obnovuje sa každé 2 sekundy. Príklad kódu môžete nájsť v prílohách ako výpis B.1.



Obr. 6.4: Dynamický graf

Záver

Cieľom tejto práce bolo vhodne uložiť dáta zo simulácie a tieto dáta zobraziť. Vybrať správny formát a spôsob ukladania. Zvoliť správny webový server a databázu na základe vypracovanej rešerše. Uložené dáta zobraziť na web stránke vo forme grafu, ktorý sa bude dynamicky načítať.

V tejto bakalárskej práci sa mi podarilo zoznámiť so softvérovým vybavením simulátoru riadenia vozidla na UAMT, čo som urobil v prvej a štvrtej kapitole. Simulácia prebiehala Unreal engine, ktorý som spomínal v kapitole 1. Opísal som jeho editor a niektoré základné komponenty a funkcie. V druhej kapitole som sa spravil stručnú rešerš ohľadom webových serverov. Na základe tejto rešerše som sa rozhodol, že si vyberiem Apache HTTP server, pretože je to open-source server, je podporovaný Windows a dokáže spracovať dynamický obsah. Ako typ databázy som si zvolil relačnú databázu, pretože mi prišlo usporiadanie v prehľadných tabuľkách ako výhodné. Z najznámejších relačných databáz som si zase vybral MySQL, pretože je open-source, je podporovaný Windows a je poskytovaná v softvérovom balíčku XAMPP.

Zvolil som dva druhy ukladania dát a to do súboru a databázy. Prišlo mi to ako najlepšie riešenie. Dáta sú "zálohované"v súbore na počítači a zároveň sa ukladajú na server, vďaka čomu je možné ich hocikedy zobraziť. Ukladanie do súboru som popísal v kapitole 5.1. Ukázal som ako som vytvoril Save menu, aký typ formátu dát som zvolil, kde dáta ukladám a taktiež aj funkciu, ktorú som vytvoril vo Visual studiu. V kapitole 5.2 som zase načrtol ukladanie do databázy. Určil som formát dát, server aj databázu, ktorú budem používať. Okrem detailného popisu ukladania som v kapitole 5 aj porovnal dáta, ktoré boli uložené na server a dáta, ktoré boli uložené do súboru. Na obrázku č. 5.16 môžete vidieť, že rozdiely v nameraných hodnotách neboli.

Pri vizualizácii som vytvoril dva spôsoby ako zobraziť dáta. Prvá možnosť je staticky, čo popisujem v kapitole 6.1. Dáta sú zobrazované v prehľadných tabuľkách, v ktorých sa je jednoduché zorientovať. Druhý spôsob je dynamicky a dáta vykresľujem v grafe. Tento spôsob som zase opísal v 6 kapitole.

Literatúra

[1] GAMESCRYE: *What is a Game Engine?* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://gamescrye.com/blog/what-is-a-game-engine/>.

[2] Studytonight: *UNDERSTANDING GAME DEVELOPMENT* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://www.studytonight.com/3d-game-engineering-with-unity/game-engine#>.

[3] Unreal Engine: *Unreal Engine* [online][cit. 20. 5. 2021]. Dostupné z URL:
<https://www.unrealengine.com/en-US/unreal>.

[4] Unreal Engine 4 Documentation: *Blueprint Overview* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/ProgrammingAndScripting/Blueprints/Overview/index.html>.

[5] Unreal Engine 4 Documentation: *Blueprint Function Libraries* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/ProgrammingAndScripting/ProgrammingWithCPP/BlueprintFunctionLibraries/index.html>.

[6] Unreal Engine 4 Documentation: *Level Editor* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/BuildingWorlds/LevelEditor/index.html>.

[7] Unreal Engine 4 Documentation: *Level Editor Toolbar* [online][cit. 21. 5. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/BuildingWorlds/LevelEditor/Toolbar/index.html>.

[8] Unreal Engine 4 Documentation: *Editor Viewports* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/BuildingWorlds/LevelEditor/Viewports/index.html>.

[9] Unreal Engine 4 Documentation: *Content Browser* [online][cit. 2. 1. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/Basics/ContentBrowser/index.html>.

[10] Unreal Engine 4 Documentation: *World Outliner* [online][cit. 21. 5. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/BuildingWorlds/LevelEditor/SceneOutliner/index.html>.

[11] Unreal Engine 4 Documentation: *Details Panel* [online][cit. 21. 5. 2021]. Dostupné z URL:
<https://docs.unrealengine.com/en-US/BuildingWorlds/LevelEditor/Details/index.html>.

[12] Tech Target: *Web Server* [online][cit. 11. 5. 2021]. Dostupné z URL:
<https://whatis.techtarget.com/definition/Web-server>.

[13] WebSupport: *Najpoužívanejšie webové servery* [online][cit. 11. 5. 2021]. Dostupné z URL:
<www.websupport.sk/>.

[14] Apache: *Apache HTTP SERVER PROJECT* [online][cit. 11. 5. 2021]. Dostupné z URL:
<https://httpd.apache.org/>.

[15] Apache Friends: *About* [online][cit. 11. 5. 2021]. Dostupné z URL:
<https://www.apachefriends.org/about.html>.

[16] TechTerms: *IIS Definition* [online][cit. 10. 5. 2021]. Dostupné z URL:
<https://techterms.com/definition/iis>.

[17] NGINX: *Welcome to NGINX Wiki!* [online][cit. 4. 1. 2021]. Dostupné z URL:
<https://www.nginx.com/resources/wiki/>.

[18] Lighttpd: *Lighttpd* [online][cit. 20. 5. 2021]. Dostupné z URL:
<https://www.lighttpd.net>.

[19] OpenResty: *OpenResty* [online][cit. 4. 1. 2021]. Dostupné z URL:
<https://openresty.org/en/>.

[20] Netcraft: *February 2021 Web Server Survey* [online][cit. 12. 5. 2021]. Dostupné z URL:
<https://news.netcraft.com/archives/2021/02/26/february-2021-web-server-survey.html>.

[21] Hacker.io: *NGINX vs Apache: Head to Head Comparison* [online][cit. 20. 5. 2021]. Dostupné z URL:
<https://hackr.io/blog/nginx-vs-apache>.

[22] Oracle Česká republika: *Databáze* [online][cit. 2. 1. 2021]. Dostupné z URL: <https://www.oracle.com/cz/database/what-is-database/#WhatIsDBMS>.

[23] MySQL: *What is MySQL?* [online][cit. 14. 5. 2021]. Dostupné z URL: <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>.

[24] MariaDB: *About MariaDB Server* [online][cit. 14. 5. 2021]. Dostupné z URL: <https://mariadb.org/about/>.

[25] PostgreSQL: *What is PostgreSQL?* [online][cit. 15. 5. 2021]. Dostupné z URL: <https://www.postgresql.org/about/>.

[26] MongoDB: *The database for modern applications* [online][cit. 14. 5. 2021]. Dostupné z URL: <https://www.mongodb.com>.

[27] DynamoDB: *Amazon DynamoDB* [online][cit. 15. 5. 2021]. Dostupné z URL: <https://aws.amazon.com/dynamodb/>.

[28] SQLite: *What Is SQLite?* [online][cit. 20. 5. 2021]. Dostupné z URL: <https://www.sqlite.org/index.html>.

[29] SQLite: *About SQLite* [online][cit. 20. 5. 2021]. Dostupné z URL: <https://www.sqlite.org/about.html>.

[30] Ansys: *Why Simulation is a Driving Force for Autonomous Vehicles* [online][cit. 20. 5. 2021]. Dostupné z URL: <https://www.ansys.com/blog/simulation-drives-autonomous-vehicles>.

[31] JSON: *Introducing JSON* [online][cit. 20. 5. 2021]. Dostupné z URL: <https://www.json.org/json-en.html>.

[32] phpMyAdmin: *About* [online][cit. 20. 5. 2021]. Dostupné z URL: <https://www.phpmyadmin.net>.

Zoznam symbolov, veličín a skratiek

UE4	Unreal engine 4
BVS	Blueprint visual scripting
www	world wide web
HTTP	Hypertext Transfer Protocol
HTML	HyperText Markup Language
BFL	Blueprint function library
IDE	Integrated development environment
IIS	Internet Information Server
C10K	concurrently handling ten thousand connections
VPS	virtuálny privátny server
URL	Uniform Resource Locator
OOP	objektovo orientované programovanie
SQL	Structured Query Language
OS	Operačný systém
DB	databáza

A Ukážka zdrojového súboru SaveFileFL.cpp

Výpis A.1: Ukážka zdrojového súboru SaveFileFL.cpp

```
#include "SaveFileFL.h"
#include "Misc/FileHelper.h"
#include "Hal/PlatformFilemanager.h"
#include "Misc/Paths.h"


bool USaveFileFL::CheckDir(FString Directory,
FString FileName)
{
  FString Dir = "";
  Dir = FPaths::ProjectSavedDir();
  Dir += "/" + Directory + "/" + FileName + ".csv";
  IPlatformFile& FileManager = FPlatformFileManager::Get()
  .GetPlatformFile();

  if (FileManager.FileExists(*Dir))
  {
    return true;
  }
  return false;
}

bool USaveFileFL::SaveData(FString Directory,
FString FileName, TArray<FString> Data)
{
  FString Dir ="";
  Dir = FPaths::ProjectSavedDir();

  IPlatformFile& PlatformFile = FPlatformFileManager::
  Get().  GetPlatformFile();
  bool AllowOverwriting = false;
  if (PlatformFile.CreateDirectoryTree(*Dir))
  {
    // Get absolute file path
    FString AbsoluteFilePath = Dir + "/" + Directory +
```

```
36      "/" + FileName + ".csv";
37      FString FinalString = "";
38
39      if (!FPaths::FileExists(*AbsoluteFilePath))
40        FinalString = "Time; Velocity; Gear; Wheel rotation;
41  Position;\n";
42
43      for (FString& Every : Data)
44      {
45        FinalString += Every;
46        FinalString += ";";
47      }
48
49      FinalString += "\n";
50      // Allow overwriting or file doesn't already exist
51      if (AllowOverwriting || !FPaths::FileExists(
52      *AbsoluteFilePath))
53      {
54        FFileHelper::SaveStringToFile(FinalString,
55        *AbsoluteFilePath);
56      }
57      else
58      {
59        FFileHelper::SaveStringToFile(FinalString,
60        *AbsoluteFilePath, FFileHelper::EEncodingOptions::
61        AutoDetect, &IFileManager::Get(), FILEWRITE_Append);
62      }
63
64    }
65    return true;
66  }
```

B Ukážka blades v Laravel

Výpis B.1: Ukážka master blade

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
␣␣␣␣initial-scale=1">
    <title>Laravel CRUD</title>
    <script src="https://ajax.googleapis.com/ajax/libs/
␣␣␣␣jquery/3.2.1/jquery.min.js"></script>
    <link href="https://cdnjs.cloudflare.com/ajax/libs/
␣␣␣␣twitter-bootstrap/4.0.0-alpha/css/bootstrap.css" rel=
    "stylesheet">
</head>
<body>
<div class="container">
    @yield('content')
</div>
</body>
</html>
```

Výpis B.2: Ukážka driver blade

```
@extends('master')
@section('content')
<div class="row">
    <div class="col-md-12">
        <br />
        <h3 align="center">Drivers</h3>
        <br />
        <div class="mt-3">
            <h1>Aktuálna simulácia</h1>
            <a class="btn␣btn-primary"
            href="{{action('DataController@chart')}}
␣␣␣␣␣␣␣␣␣␣␣␣">Zobraziť dáta</a>
        </div>
```

```
14          <table class="table␣table-bordered␣table-striped">
15              <tr>
16                  <th>First Name</th>
17                  <th>Last Name</th>
18                  <th>Select</th>
19              </tr>
20              @foreach($driver as $row)
21                  <tr>
22                      <td>{{$row['first_name']}}</td>
23                      <td>{{$row['last_name']}}</td>
24                      <td><a href="{{action
25  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣('SimulationController@show',$row
26  ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣['id'])}}" class="btn␣btn-warning">
27                      Select</a></td>
28                  </tr>
29              @endforeach
30          </table>
31      </div>
32  </div>
33  @endsection
```

C Ukážka kódu v JavaScript

Výpis C.1: Ukážka kódu v JavaScript

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Actual data</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js">
    </script>
    <script src="https://code.jquery.com/jquery-3.1.1.min
␣␣␣␣.js"></script>
</head>
<body>
<div class="container">
    <h1 style="text-align:␣center;">Actual data</h1>
    <br>
</div>
<div>
    <canvas id="myChart1"  height="280" width="600">
    </canvas>
</div>

<script>
    // === include 'setup' then 'config' above ===
    const labels = [

    ];
    const data = {
        labels: labels,
        datasets: [{
            label: 'velocity',
            backgroundColor: 'rgb(241,0,56)',
            borderColor: 'rgb(241,0,56)',
            data: [],
        }]
    };

```

```
36      const config = {
37          type: 'line',
38          data,
39          options: {}
40      };
41
42      var myChart1 = new Chart(
43          document.getElementById('myChart1'),
44          config
45      );
46
47      function aktualizovat(chart) {
48          $.getJSON("http://localhost:8000/api/data/",
49          function (result) {
50              console.log(result);
51              let label=[];
52              let data=[];
53              $.each(result, function (i, field) {
54                  label.push(field["time"]);
55                  data.push(field["velocity"]);
56
57              });
58              chart.data.labels=label;
59              chart.data.datasets[0].data=data;
60              chart.update();
61          });
62
63      }
64      setInterval(function(){aktualizovat(myChart1);},2000);
65
66  </script>
67  </body>
68  </html>
```

D Obsah priloženého média

Na priloženom médiu sa nachádza samotná bakalárska práca vo formáte .pdf. Rovnako tak aj súbory so všetkými modelmi, kontorlérmi a bladami navrhnutý v Laraveli.

```
/ ..... koreňový adresár priloženého média
└── Bakalárska práca
    ├── Bakalárska práca-Dominik Viater.pdf
    ├── Controllers
    │   ├── DriverController.php
    │   ├── SimulationController.php
    │   └── DataController.php
    ├── Models
    │   ├── Driver.php
    │   ├── Simulation.php
    │   └── Data.php
    ├── views
    │   ├── master.blade.php
    │   ├── driver.blade.php
    │   ├── simulation.blade.php
    │   ├── data.blade.php
    │   └── chart.blade.php
    └── .env
```